Mid-Infrared Data Analysis Software 2010

A Matlab Interface for

FTIR Spectroscopy Analysis

User's Guide

Elise Normand

January 10, 2011

Contents

List of Figures

List of Tables

1 Purpose

The *M*id-*I*nfrared *D*ata *A*nalysis *S*oftware 2010 (*MIDAS 2010*) was created for the specific use on the Mid-Infrared Spectromicroscopy beamline at the Canadian Light Source (CLS) at the University of Saskatchewan. Please refer to the CLS website for specifics detailing the end stations, techniques, and optics for the Mid-IR beamline (`http://midir.lightsource.ca`).

MIDAS 2010 can be used as a verification tool. The current software program on the Mid-IR beamline used to analyze the data from the Bruker FTIR spectrometer and Hyperion microscope occasionally encounters some plotting problems. MIDAS 2010 was created in an effort to avoid these plotting issues as well as to ease selecting specific parts of the data for analysis.

Aside from some of the regular Matlab tools and utilities, MIDAS 2010 can help the user load and prepare the data, compute the fast Fourier transform and cross correlation of the selected data, and graphically display the results in a variety of ways. Additionally, MIDAS 2010 has the ability of applying various types of filters to the data as to eliminate high frequency noise.

MIDAS 2010 is the second version of this program, the first of which was simply named MIDAS.

2 System Requirements

In order for MIDAS 2010 to work properly, the following software must be installed on the user's computer:

- *Matlab*: Version 7.6.0 (R2008a) - Must include the **Image Processing Toolbox** (Note that MIDAS 2010 has not been tested using previous versions of Matlab)
- Any text editor, such as *Notepad*: Version 5.1 - Service Pack 3
- *OPUS*: Version 6.0 or better

3 Installations

Installing MIDAS 2010 is very easy. Simply save the *MIDAS2010* folder in an appropriate location. This folder must be accessed by Matlab's Current Directory each time MIDAS 2010 is used. The following Figure and M files should be within the MIDAS2010 folder:

applyFilter.m
averageDynamic_Final.m
averageDynamic.m
averageKernel.m
bandpassfilter.m
dftfilt.m
dynamicTracesExist.m
EFilter.m
exportFigure12_TXT.m
exportFigure12_XLS.m
ExportPassfile.m
exportPlot12_TXT.m
exportPlot12_XLS.m
exportPlotAB_TXT.m
exportPlotAB_XLS.m
fastbilateral.m
fft2dcorr.m
fft2dcorrelation.m
Figure1GUI_2010.m
Figure2GUI_2010.m

FilterGUI_2010.m
findRandomColour.m
gaussianfilter.m
getColourMap.m
highboostfilter.m
highEFilter.m
highpassfilter.m
loadfile.m
LoadGUI_2010.m
lowpassfilter.m
MainGUI_2010.m
MIDAS2010.m
NewLabels.m
NewTitle.m
OpenGUI.m
OpenPlotGUI.m
OpenWorkspace.m
opus_read.m
paddedsize.m
plot2d_function.m
plotspectra_function.m
plotspectra_timeselected_compare.m
plotspectra_timeselected.m
plotspectra_wavselected_compare.m
plotspectra_wavselected.m
removeStatic.m
saveButton.m
saveFigure12.m
saveFigureAB.m
savePic.m
SaveWorkspace.m
timestep.m
updateFigure1.m
updateFigure2.m
Figure1GUI_2010.fig
Figure2GUI_2010.fig
FilterGUI_2010.fig
LoadGUI_2010.fig
MainGUI_2010.fig
MIDAS2010.fig
NewLabels.fig
NewTitle.fig
MIDAS2010_UserManual.pdf

An example text file, named *RodCellData.txt*, is also included within the MIDAS2010 folder.

4 Getting Started

4.1 Data Requirements

Before any data can be loaded using MIDAS 2010, the user must save the data as a tab delimited text file with extension *.txt*. The data saved in a text

file must be a matrix with $[m\ n]$ dimensions. The first column of the matrix must contain the variables, such as wavenumbers, while the remaining columns contain samples, like absorbance units. The dynamic variable, such as temperature or time, is incremented by a constant amount δ between each column starting from column 2 to n. That is, each row contains the wavenumber in the first cell and the various absorbance data are in the remaining cells. Temperature (or time) is incremented between each column starting with temperature (or time) $T = 0$ in column 2, $T = \delta$ in column 3, ..., and $T = (n-2) * \delta$ in column n.

As an example, for a temperature (or time) dependent IR spectra, meaning that the dynamic variable is temperature (or time), the first column should contain the wavenumbers and the remaining columns should contain the corresponding absorbance units. The absorbance values in the cells represent the average absorbance over the temperature (or time) increment since temperature (or time) is augmented by a constant amount δ between columns 2 through n. Visually, the data matrix should look similar to Table 1.

1400.2	-0.000103	-0.000257	0.000090
1398.3	0.000198	0.000513	0.000222
1396.4	0.000097	-0.000365	-0.000189

Table 1: Example Data Matrix

4.2 Step to Saving Data Matrices

A simple way to save the data matrix in the text file with the appropriate requirements is to:

1. Open the Report Window in OPUS. The Report Window can be accessed through the Window tab menu.

$$Window \;-->>\; New\ Report\ Window$$

2. Select and copy all the data. If there is a lot of data, there may not be enough memory space to copy all the data at once. In this case, the data must be copied in intervals. Press and hold the *SHIFT* key to ease selecting multiple rows of data at once.

3. Past the copied data into a text file. Notepad is the suggested text editor. Do not modify the pasted data. It will look very messy because the columns and rows will no longer be properly aligned. Although this is not too appealing to us, the computer knows which data is in each row and column. The data should already be tab delimitated; this should be apparent.

4. Save the text file in an appropriate location. Make sure the file is saved with the *.txt* extension.

5. Open Matlab and set the *MIDAS2010* folder to the Current Directory. The *Browse For Folder* button has three small dots on it and is located

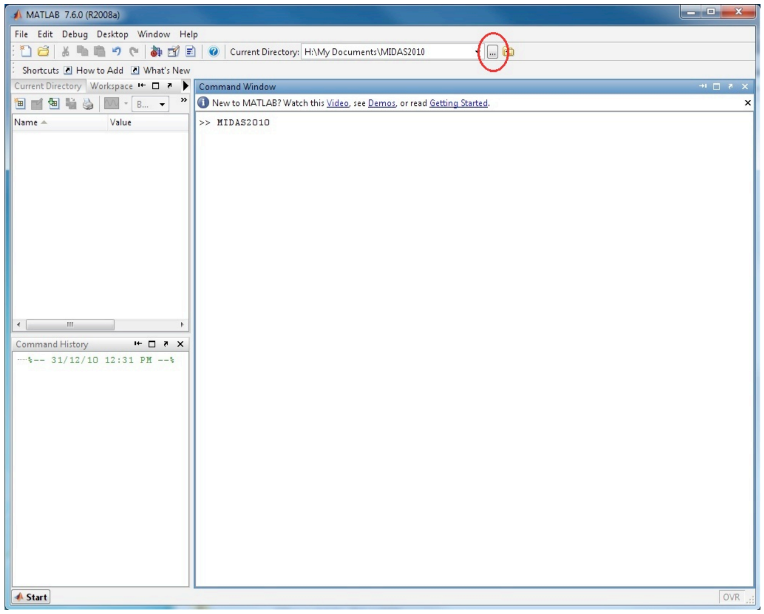

Figure 1: MATLAB Window

on the top of the Matlab window; it is shown in Figure 2. At this point, MIDAS 2010 is ready to analyze the data in the text file.

MIDAS 2010 can now be used to analyze the spectroscopy data. To run the software, type *MIDAS2010* (case sensitive) in Matlab's Command Window.

$$>> \ MIDAS2010$$

The MIDAS 2010 window, shown in Figure 3, will appear. Note that Matlab will show a *busy* signal on the lower left hand corner of the Matlab window whenever it is computing an operation.

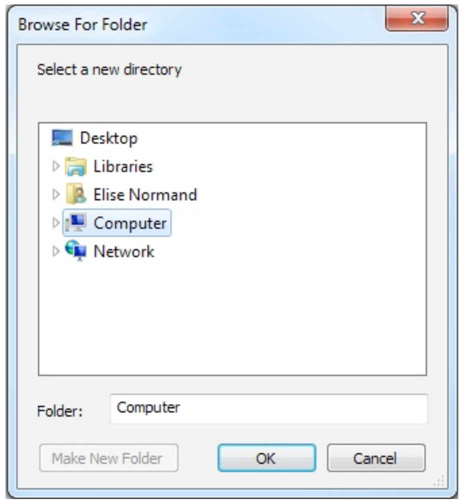

Figure 2: Browse For Folder Window

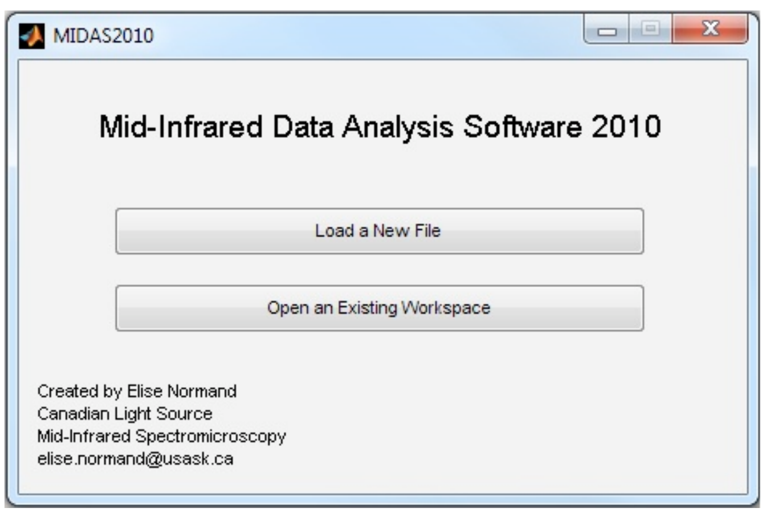

Figure 3: MIDAS 2010 Window

There are two ways to pull up an interface using MIDAS 2010. The first involves loading a new data file, such as a data text file, and the second involves opening a previously saved workspace. To load a new set of data, press the *Load a New File* button. The loading window shown in Figure 4 will appear. To open previously saved work, press the *Open an Existing Workspace* button. A dialogue box will appear asking the user to find and select the workspace figure. Once the figure is selected, the workspace window will appear as it was last saved.

5 Loading Data Interface

The loading window, named *LoadGUI_2010*, appears upon request from the MIDAS 2010 window. Note that only one loading interface can appear at a time; meaning only one text file can be loaded into MIDAS 2010 at a time[1].

5.1 Selecting a Data File

A file must be selected before any other adjustments can be made. When the *Select File* button is pressed, a dialog box will appear asking the user to select the desired text file. After the file is selected, the file path is shown on the loading window.

[1] In future versions of MIDAS, the programmer should allow for more than one file to be loaded at a time. Check out *singleton*, *guide*, and *property inspector* in Matlab.

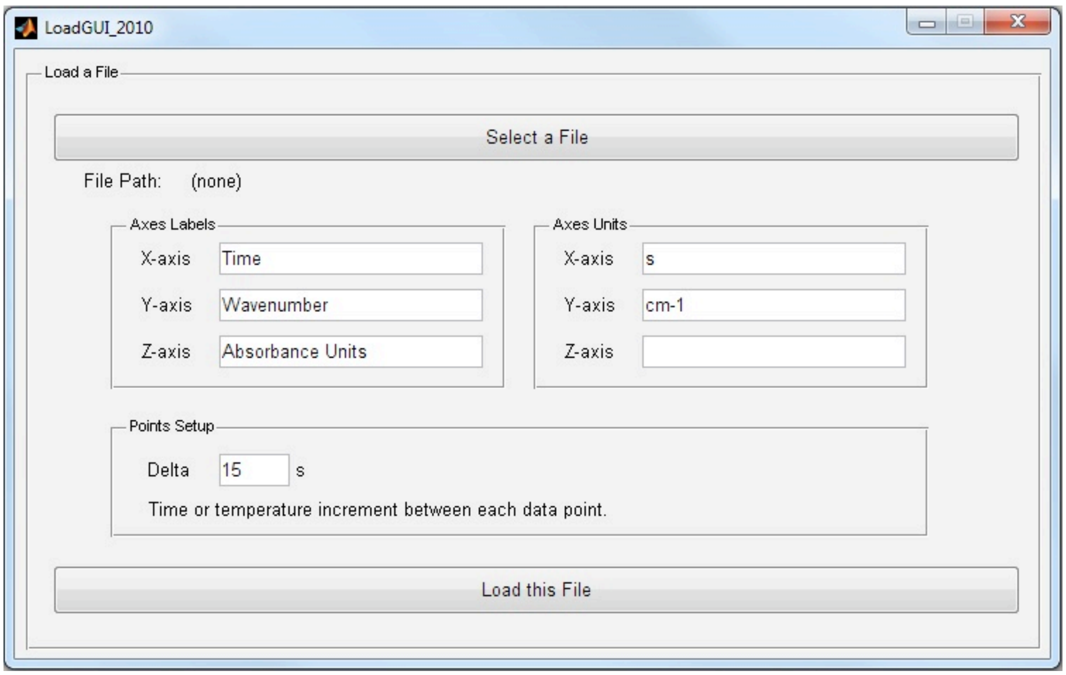

Figure 4: Load a File Interface

5.2 Labeling Axes and Units

The X, Y, and Z axes corresponding to the spectra plot can be named at this stage as well as their corresponding units. When plotted, the X, Y, and Z axes represent the temperature (or time), wavenumber, and absorbance units, respectively.

5.3 Points Setup

In order for MIDAS 2010 to display the correct results, it is necessary to set *Delta* appropriately. *Delta* is the constant δ temperature (or time) increment between the columns of the data matrix. A point is recorded every δ degrees (or seconds). It is assumed that δ is constant for the data set. Since δ represents an increment, it must be positive and larger than zero and it must have units of degrees (or seconds).

6 Main Interface

MIDAS' main window is shown in Figure 5. Notice the four plotting areas named *Plot A*, *Plot B*, *Plot 1*, and *Plot 2*. Each plotting region was designed to help the user view the data in different ways. They are explained in the following sections.

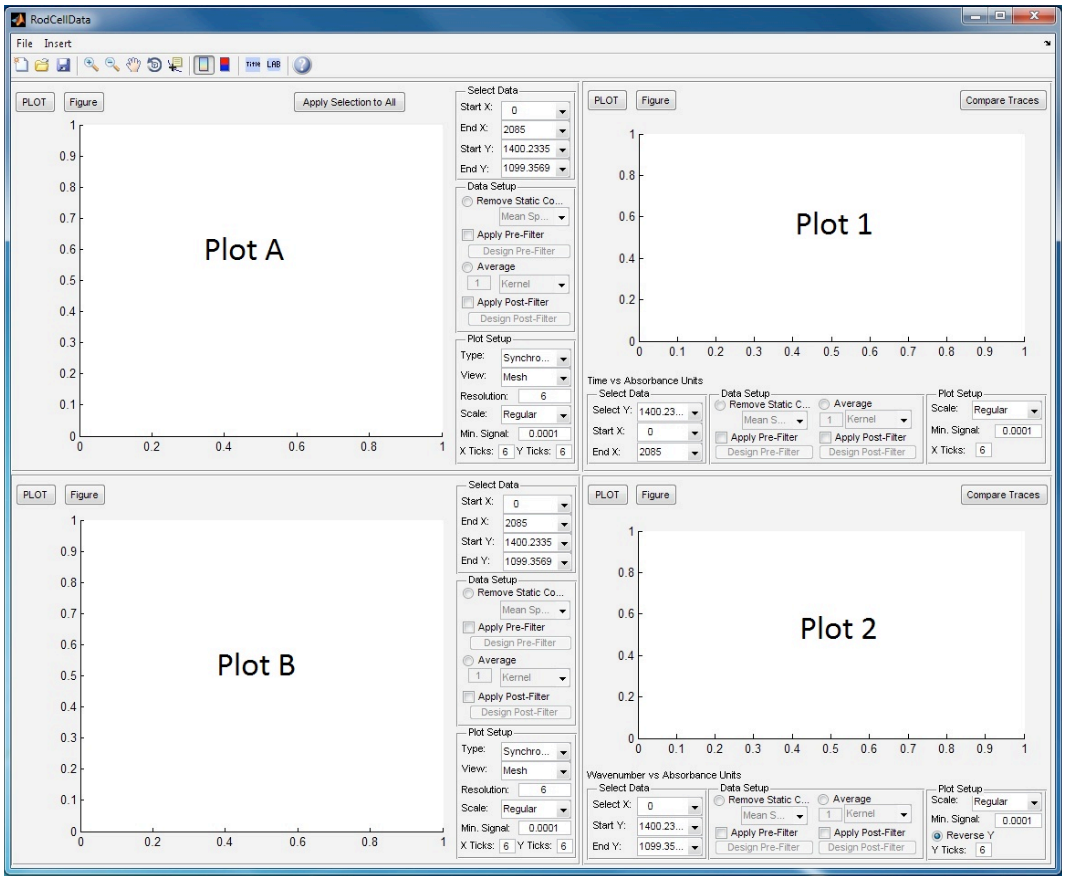

Figure 5: Main Window

6.1 Plots A and B

Plots A and B are the top and bottom left hand plots in the main window and function identically. The purpose of these similar plots is to ease comparing the same data set while varying the analytical tools. Altering the *method* in which the data is processed or analyzed and comparing the results can be advantageous for the user.

The primary plots used in MIDAS 2010 are Plots A and B. They have the option of plotting the *synchronous*, *asynchronous*, *phase*, or *spectra* graphs. To procure these graphs, the Fourier transform and 2D correlation of the data matrix are computed and result in a matrix of complex numbers. The *synchronous* graph is obtained from the real components while the *asynchronous* is obtained from the imaginary components. The phase between the real and imaginary components forms the *phase* graph. The *spectra* graph plots the original data before the Fourier transform and 2D correlation are performed. As such, the X, Y, and Z axes represent the temperature (or time), wavenumber, and absorbance units, respectively, as specified when the data was initially loaded with the loading window.

6.2 Plots 1 and 2

Plots 1 and 2 plot orthogonal cross sections of the spectra data. Plot 1 is the top right hand plot and Plot 2 is the lower right hand plot in the main window. Plot 1 graphs the cross section along a specified wavenumber.

That is, it plots the temperature (or time) versus absorbance units along a specific energy; it graphs X versus Z for a selected Y. Plot 2 graphs the perpendicular cross section which is along a specified temperature (or time). That is, it plots the wavenumber versus the absorbance units along a specific temperature (or time); it graphs Y versus Z for a selected X.

7 Analytical Tools

The following plotting options are available to customize the method in which the data is analyzed and viewed. Some options are strictly available to Plots 1 and 2.

7.1 Selecting Data

7.1.1 Start and End X, Y

The drop down menus in this section allow the user to define the X and Y plotting ranges. *Start X* and *End X* correspond to the start and end dynamic variables (as in temperature or time) while *Start Y* and *End Y* correspond to the start and end wavenumbers. Recall that the X, Y, and Z axes represent the temperature (or time), wavenumber, and absorbance units, respectively, from the spectra plot as specified when the data was initially loaded with the loading window.

The *Start X* and *End X* values in the drop down menus take into ac-

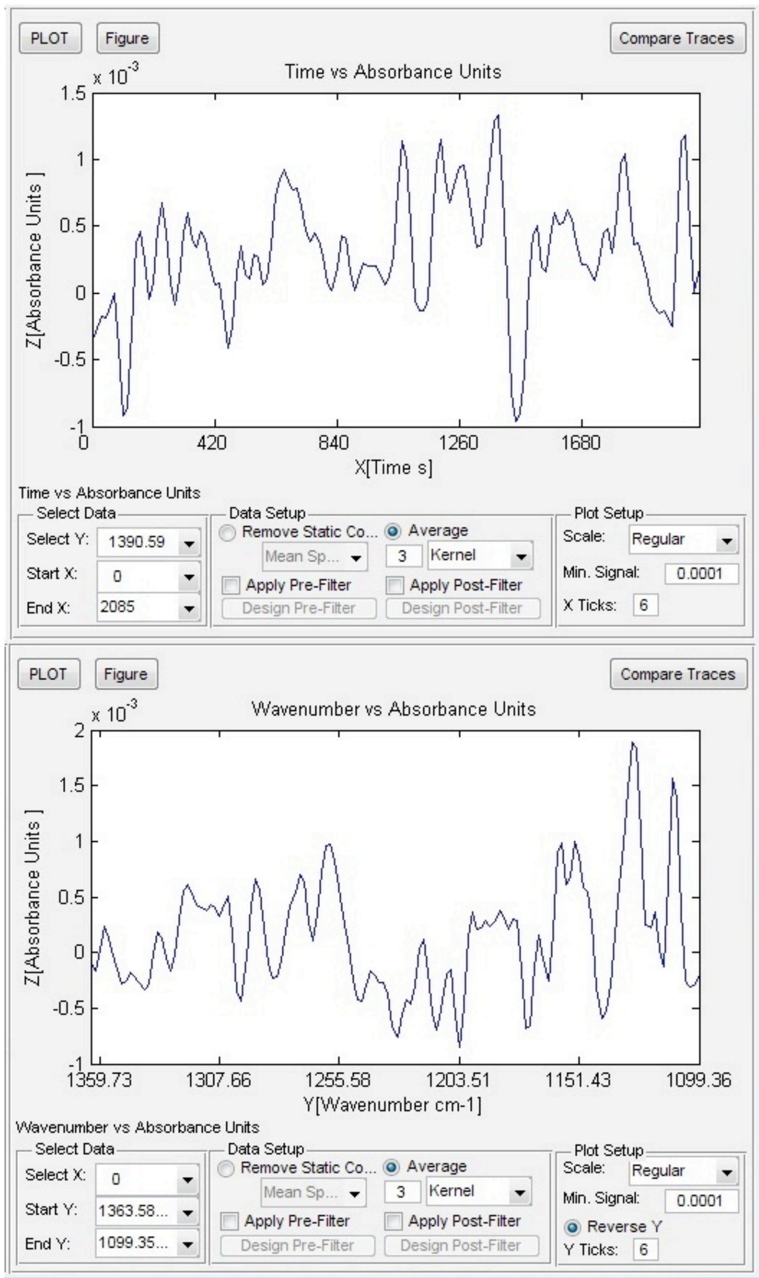

Figure 6: Plots 1 and 2

count δ from the loading window. Recall that *delta* is the constant δ temperature (or time) increment between the columns of the data matrix. The *Start Y* and *End Y* values in the drop down menus are taken directly from the input data matrix; it is the first column from the text file.

7.1.2 Select Y and X

Recall that Plots 1 and 2 plot orthogonal cross sections of the spectra data. Plot 1 plots the temperature (or time) versus absorbance units along a specific energy; it graphs X versus Z for a selected Y. Therefore, the user can select the specific energy using the *Select Y* drop down menu.

Similarly, Plot 2 plots the wavenumber versus the absorbance units along a specific temperature (or time); it graphs Y versus Z for a selected X. The user can select the specific temperature (or time) using the *Select X* drop down menu.

7.2 Data Setup

7.2.1 Remove Static Component

There are two types of static components the user can remove from their data: the mean spectrum and steady state component.

Mean Spectrum: The mean spectrum is defined as the average trace across the dynamic variable. When the input data is loaded from the text file, each column is treated as a vector and the average value is returned for

each vector. This forms a mean trace along the dynamic variable. That is, for each δ temperature (or time) increment, the average over all wavenumbers is computed, and together these averages form a mean trace along the temperature (or time) variable. Removing the mean spectrum from the data effectively subtracts the mean value from the respective column. Therefore, the resulting data represents the change relative to the mean trace rather than the absolute data. Keeping the mean spectrum leaves the data unaltered. In this case, the data shown in the graph is then the absolute data and not the relative change from the mean trace.

Steady State: The steady state component is defined as the very first column of the input data matrix which corresponds to the initial temperature (or zero time). It is essentially the DC component. Removing the steady state component effectively subtracts the first column from each column of the data matrix. Therefore, the resulting data represents the change relative to the DC component rather than the absolute data. Removing the steady state component automatically sets the first column of data to zero since it cancels out itself through the subtraction. Keeping the steady state component means the DC component is left in the data. In this case, the data shown in the graph is then the absolute data and not the relative change over the dynamic variable.

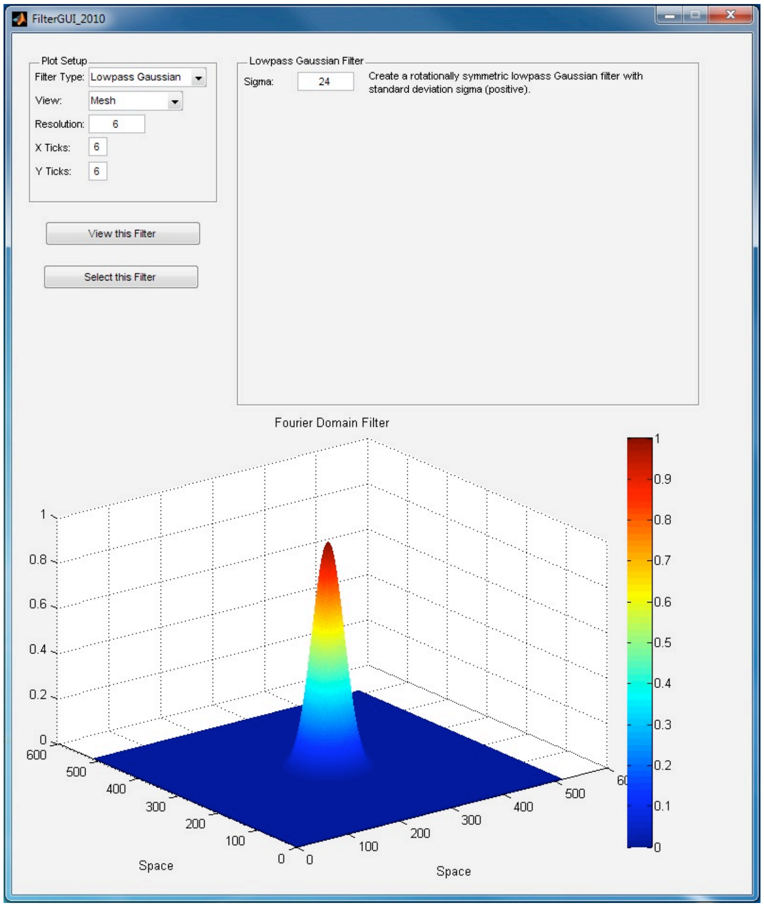

Figure 7: Filter Interface: Lowpass Gaussian

7.2.2 Apply Pre- and Post-Filter

Selecting the *Apply Pre-Filter* or *Apply Post-Filter* option will bring up the filtering interface shown in Figure 7. The operations of the filtering window are explained in Section 8 below. When the *Apply Filter* option is selected, the current or last filter applied to the data is shown on the interface.

7.2.3 Average

There are two types of averaging available in MIDAS 2010, and both produce drastically different results. They are the kernel and dynamic averages.

Kernel Average: Selecting kernel average smoothes the data along the dynamic variable using an averaging kernel. The kernel behaves as a sliding window where it is centered over each cell in the data matrix, and the sum of products of the kernel coefficients and the underlying cells is computed[3]. The size of the data is extended by mirror-reflecting the data across its border (at the edges of the data matrix).

If *Average* is set to some number ϵ, then the kernel has dimensions $[1\ \epsilon]$. Note that ϵ must be an odd number to assure the window has a center cell. The kernel coefficients are modified to take an average over the dynamic variable (temperature or time) only.

MIDAS 2010 uses a Matlab function called *fspecial*, which belongs to the Image Processing Toolbox, to design the kernel. *fspecial* creates an average filter using[2]

$$\frac{\mathrm{ones}(1,\epsilon)}{1*\epsilon}$$

The kernels with $\epsilon = 3$ and $\epsilon = 5$ are shown in Tables 2 and 3, respectively.

Dynamic Average: When dynamic averaging is selected, the mean between data points along the dynamic variable (temperature or time) is

[2]Description is from Matlab Help.

0.3333	0.3333	0.3333

Table 2: Kernel Average: $\epsilon = 3$

0.2	0.2	0.2	0.2	0.2

Table 3: Kernel Average: $\epsilon = 5$

taken. If *Average* is set to some number ϵ, for example, then the average of the first ϵ data points in a given row is taken and placed in a cell. The following ϵ data points from the same row are then averaged and placed in the following cell. This process is repeated until the end of the row is reached or until there are less than ϵ data points left in the row and an average cannot be taken. Thus, if the original data matrix has dimensions $[a\ b]$, namely b points corresponding to the dynamic variable, and a dynamic average is taken every ϵ points, then the resulting data matrix has dimensions $[a\ b/\epsilon]$. Of course, an average of 1 does not modify the original matrix. A visual example of dynamic averaging is shown in Figure 8.

MIDAS 2010 re-computes the temperature (or time) increment between successive points in the data matrix using δ which was defined in the loading window and in Section 5.3. If the user selects *Dynamic Average* with $\epsilon > 1$, then the resulting data matrix is compressed along the dynamic (temperature or time) variable; the increment between the columns of the data matrix is no longer δ. Rather, the increment between the columns of the compressed data matrix is now $\delta * \epsilon$. The lists in the *Start X* and *End X*

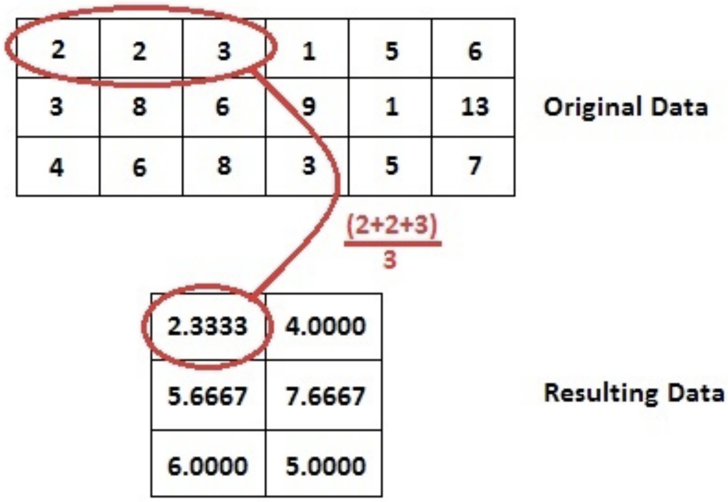

Figure 8: Dynamic Average Example

drop down menus are modified accordingly.

7.3 Plot Setup

7.3.1 Type

There are four types of graphs for Plots A and B, as previously mentioned.

- *Synchronous*
- *Asynchronous*
- *Phase*
- *Spectra*

To procure these graphs, the Fourier transform and 2D correlation of the data matrix are computed and result in a matrix of complex numbers. The *synchronous* graph is obtained from the real components while the *asynchronous* is obtained from the imaginary components. The phase between the real and imaginary components forms the *phase* graph. The *spectra* graph plots the original data before the Fourier transform and 2D correlation are performed. As such, the X, Y, and Z axes represent the temperature (or time), wavenumber, and absorbance units, respectively, as specified when the data was initially loaded with the loading window.

7.3.2 View

For Plots A and B, there are six ways to view the desired graphs[3]:

- *Mesh* - Wireframe parametric surface graded with colour.
- *Contour* - Two dimensional isopleth or topographic map with contour levels graded with colour.
- *Mesh and Contour* - Wireframe parametric surface with a contour plot beneath the mesh, both are graded with colour.
- *3D Contour* - Three dimensional contour plot graded with colour.
- *Contour Function* - Filled two-dimensional contour plot using constant colours between ioslines.

[3]Viewing descriptions are from Matlab Help.

- *Waterfall* - Similar to mesh but with a waterfall effect.

Figures 9 and 10 show the mesh and contour plots of two identical synchronous and asynchronous data sets.

Note: *Resolution* is only used for contour and contour-related plots. It determines the number of levels or isolines.

7.3.3 Scale and Minimum Signal

There are three scale options:

- *Regular*
- *Variance*
- *Range*

Both the variance and range scale options depend on the *Minimum Scale* input.

Variance: The correlation intensities between two large peaks that are only slightly correlated may be larger than the correlated intensity between two smaller peaks that are highly correlated. The variance scaling on the correlation intensities is evaluated for the univariate case, meaning row by row in the data matrix. Scaling the data by the Euclidean length, as in the square root of the sum of the squared intensities for each resolution element, gives consistent results with statistical correlation[4].

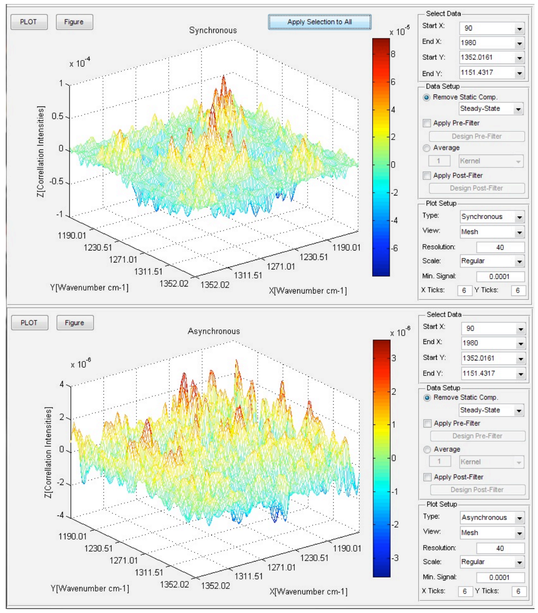

Figure 9: Mesh Plots: Synchronous and Asynchronous Plots

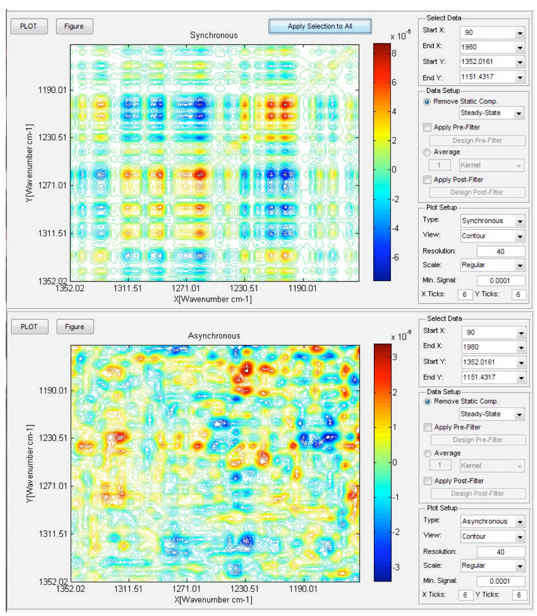

Figure 10: Contour Plots: Synchronous and Asynchronous Plots

Essentially, when the variance scale option is selected, the maximum absolute value across the dynamic variable (temperature or time) is found and compared to the *Minimum Signal* value. If the maximum absolute value is found to be larger than the minimum signal value, then the data across the dynamic variable for the particular wavenumber is scaled by the Euclidean length.

Range: Range scaling divides each function by the range as defined by the maximum function value minus the minimum function value. This method of scaling has little effect on the synchronous and asynchronous components. When the functions decay back to the baseline, range scaling has some effect. The slope of the synchronous component is much less and the maximum in the asynchronous component has disappeared[4].

When the range scaling option is selected, the maximum absolute value across the dynamic variable is compared to the *Minimum Signal* value. If the maximum absolute value is found to be larger than the minimum signal value, then the data across the dynamic variable for the particular wavenumber is divided by the difference between the maximum absolute and the minimum signal values.

Regular: Regular scaling applies neither variance nor range scaling methods to the data.

7.3.4 Axes Ticks

The user has the option of determining the number of tick marks on both the X and Y axes. The X and Y axes refer to the temperature (or time) and wavenumber axes, respectively, from the spectra plot. Thus, when the synchronous, asynchronous, and phase plots are viewed, only the *Y axis* ticks input is used since these plots have wavnumbers plotted along two axes. Note that the input values must be integers larger than two.

7.3.5 Reverse Y

Some users find it convenient to view the wavenumbers ascend on the Y axis in Plot 2 while others prefer to view them descend. Selecting the *Reverse Y* option graphs the magnitude of the wavenumbers in a decreasing order. Deselecting this option simply reflects the graph about the Z axis.

8 Filtering Interface

Selecting the *Apply Pre-Filter* or *Apply Post-Filter* option will bring up the filtering interface shown in Figure 7. When the *Apply Filter* option is selected, the current or last filter applied to the data is shown on the interface. The filters are applied to the data in the Fourier domain.

8.1 Periods in Time and Fourier Domains

The period in the Fourier domain along the *U* and *V* axes are equal to the number of samples in the *X* and *Y* directions of the input data in the time domain, respectively. If the number of samples are nx and ny respectively, then the range $-nx/2$ to $+nx/2$ on the *U* axis comprises one period in the *U* direction, while the range $-ny/2$ to $+ny/2$ on the *V* axis comprises one period in the *V* direction[1].

8.2 Frequency Domain Filtering

The filters are applied to the data in the Fourier domain. The following list enumerates the basic steps[2] taken to achieve filtering in the Fourier domain.

1. Typical discrete Fourier transforms (DFT) are infinitely periodic. This causes problems when multiplying DFT's together. If periods overlap, the product is subject to *wraparound error*. Padding both the data $f(x, y)$ and the filter $h(x, y)$ ensures they both have the same periods in the frequency domain. If $f(x, y)$ and $h(x, y)$ are of size $[A\ B]$ and $[C\ D]$ respectively, we can form two padded functions, both of size $[P\ Q]$ by padding f and h with zeros. Wraparound error is avoided by choosing

$$P \geq A + C - 1$$

$$Q \geq B + D - 1$$

Therefore, obtain the padding parameters P and Q.

2. Compute the two-dimensional fast Fourier transform of the data with the padding to obtain the DFT of the data.

3. Multiply the DFT by the Fourier-domain filter $H(x, y)$:

$$G = H. * DFT$$

4. Obtain the real part of the two-dimensional inverse discrete Fourier transform of the resulting product $G(x, y)$. The real part $g(x, y)$ is now in the time domain due to the inverse Fourier transform.

5. Crop the result to the original size to eliminate extra padding.

8.3 Plot Setup

The following plotting options are available to customize the method in which the filter is viewed.

8.3.1 Filter Type

There are eight types of filters the user can apply to the data:

- *Lowpass Gaussian Filter*
- *Lowpass Butterworth Filter*

- *Lowpass Elliptical Filter*
- *Highpass Gaussian Filter*
- *Highpass Butterworth Filter*
- *Highpass Elliptical Filter*
- *Bandpass Filter*
- *Highboost Filter*

These filters are explained in the following sections.

8.3.2 View

There are six ways to view the desired filter[4]:

- *Mesh* - Wireframe parametric surface graded with colour.
- *Contour* - Two dimensional isopleth or topographic map with contour levels graded with colour.
- *Mesh and Contour* - Wireframe parametric surface with a contour plot beneath the mesh, both are graded with colour.
- *3D Contour* - Three dimensional contour plot graded with colour.
- *Contour Function* - Filled two-dimensional contour plot using constant colours between ioslines.

[4]Viewing descriptions are from Matlab Help.

- *Waterfall* - Similar to mesh but with a waterfall effect.

These are identical to those described in Section 7.3.2.

8.4 Filtering Buttons

When a filter is designed by manipulating the filter parameters, the following buttons may be used to confirm or cancel the filter design.

View this Filter: When a filter is designed with new parameters, the filter will be compiled and plotted when the user presses the *View this Filter* button.

Select this Filter: When the user presses the *Select this Filter* button, the user confirms the filter designed as per the indicated parameters.

Cancel Filter Design: When the user presses the X on the top right hand corner of the filtering interface, the user cancels the design and no filter is applied to the data. Notice how the *Apply Filter* checkbox is not checked.

8.4.1 Parameters Versus Plotting Priority

Note that if any filter parameter is modified and if the user does not press the *View this Filter* button before pressing the *Select this Filter* button (meaning the view of the filter was not updated to account for the change in parameters), then the confirmed filter is designed as per the indicated parameters, not as was shown in the figure. A change in parameters always trumps the view of the plotted filter.

8.5 Lowpass Gaussian Filter

When the lowpass butterowrth filter is selected, the window in Figure 7 is displayed. MIDAS 2010 uses a Matlab function called *fspecial*, which belongs to the Image Processing Toolbox. This function returns a rotationally symmetric Gaussian lowpass filter with standard deviation σ (positive). *fspecial* creates a Gaussian filter using[5]

$$h_g(n_1, n_2) = exp\frac{-(n_1^2 + n_2^2)}{2\sigma^2}$$

$$h(n_1, n_2) = \frac{h_g(n_1, n_2)}{\sum_{n_1} \sum_{n_2} h_g}$$

Sigma: The variable *sigma* is the standard deviation σ.

8.6 Lowpass Butterworth Filter

When the lowpass butterowrth filter is selected, the window in Figure 11 is displayed. The user can design a rotationally symmetric lowpass butterworth filter[8] using the following variables:

Cutoff: The *cutoff* variable is the cutoff frequency of the filter and must lie within the range 0 to 0.5.

Order: The variable *order* is the order of the filter. The higher the order, the sharper the transition. The order must be an integer greater than 1.

[5] Description is from Matlab Help.

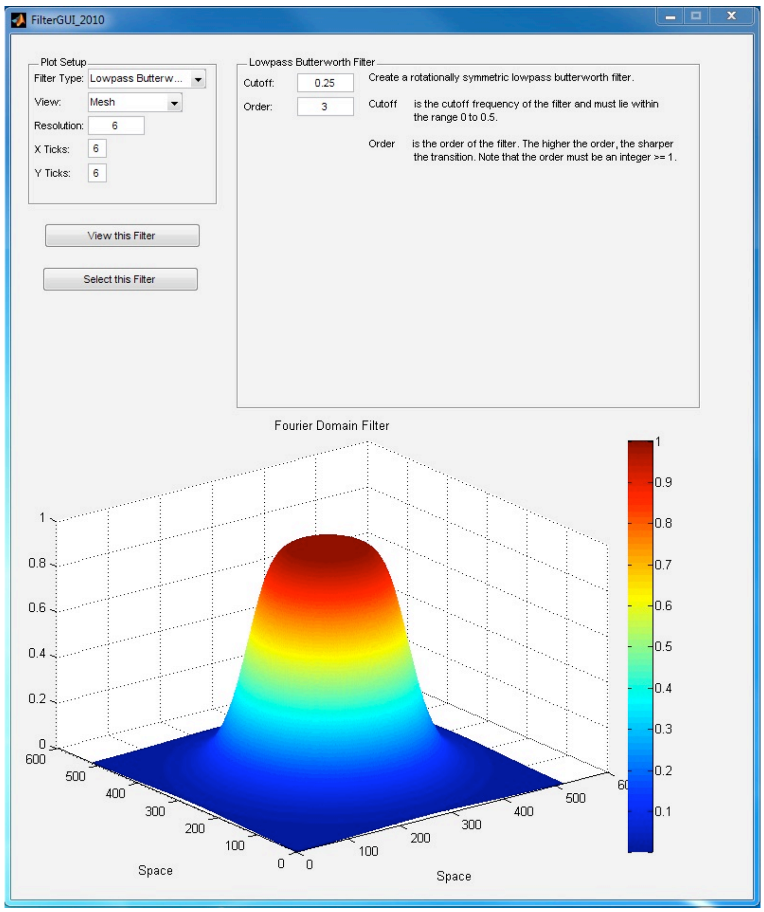

Figure 11: Filter Interface: Lowpass Butterworth

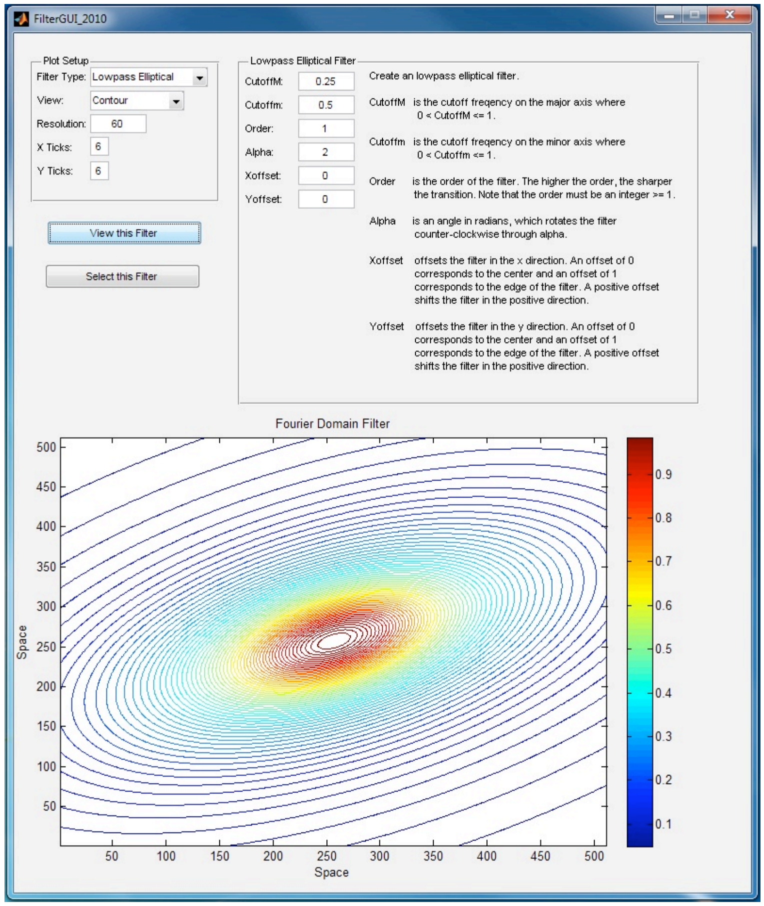

Figure 12: Filter Interface: Lowpass Elliptical

8.7 Lowpass Elliptical Filter

When the lowpass elliptical filter is selected, the window in Figure 12 is displayed. The user can design an nth order elliptical lowpass digital butterworth filter[9] using the following variables:

CutoffM and Cuttoffm: The variables *cutoffM* and *cutoffm* are the cutoff frequencies on the major and minor axes respectively, and lie within the

range $0 < \text{cutoff} \leq 1$.

Order: *Order* is the order of the filter. The higher the order, the sharper the transition. The order must be an integer greater than 1.

Alpha: The variable *alpha* is an angle in radians, which rotates the filter counter-clockwise through alpha. Obviously and angle of zero does not rotate the filter.

Xoffset and Yoffset: The *Xoffset* and *Yoffset* variables offset the filter in the x and y directions respectively. An offset of 0 corresponds to the center of the filter, and an offset of 1 corresponds to the edge of the filter. A positive offset shifts the filter in the positive direction while a negative offset shifts the filter in the negative direction.

8.7.1 Designing Gaussian and Butterworth Filters Using the Elliptical Filter

A *Gaussian* filter can easily be designed with the *Elliptical* filter if $\text{cutoffM} = \text{cutoffm} = 0.5$, $\text{order} = 1$, and $\text{alpha} = \text{Xoffset} = \text{Yoffset} = 0$. Similarly, a *butterworth* filter can be designed with the same parameters, but with $\text{order} > 1$.

8.8 Highpass Gaussian Filter

When the highpass Gaussian filter is selected, the window in Figure 13 is displayed. The highpass Gaussian filter uses the lowpass Gaussian filter

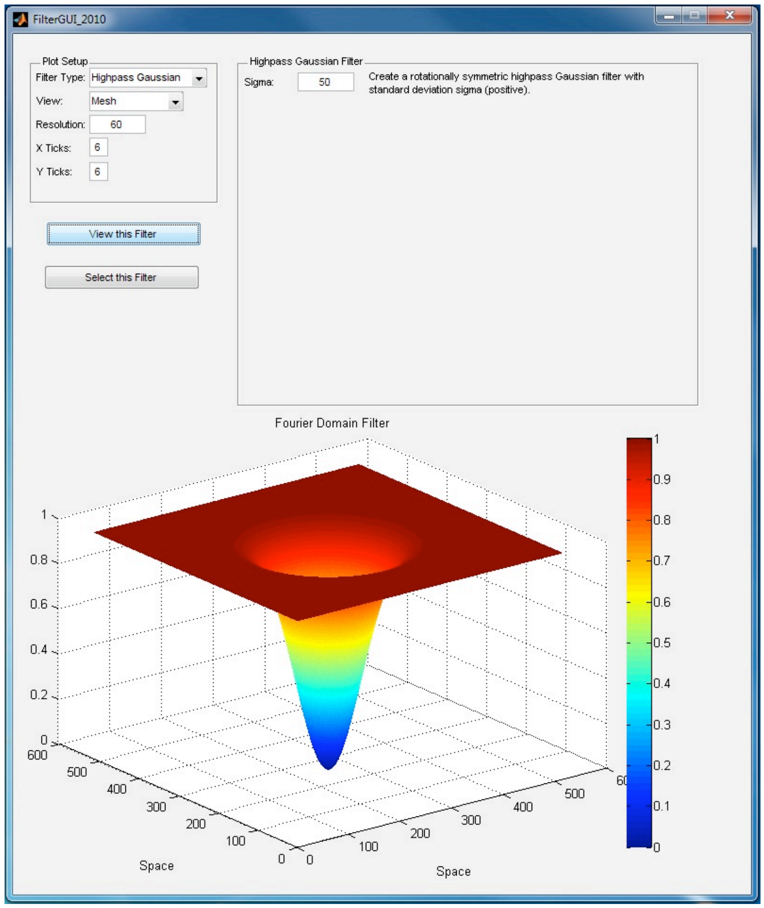

Figure 13: Filter Interface: Highpass Gaussian

techniques described in Section 8.5. However, the highpass filter is defined as a function of the lowpass filter:

$$h_{highpass}(x, y) = 1 - h_{lowpass}(x, y)$$

Sigma: The variable *sigma* is the standard deviation σ.

8.9 Highpass Butterworth Filter

When the highpass butterworth filter is selected, the window in Figure 14 is displayed. The highpass butterworth filter[7] uses the lowpass butterworth filter techniques described in Section 8.6. However, the highpass filter is defined as a function of the lowpass filter:

$$h_{highpass}(x, y) = 1 - h_{lowpass}(x, y)$$

Cutoff: The *cutoff* variable is the cutoff frequency of the filter and must lie within the range 0 to 0.5.

Order: The variable *order* is the order of the filter. The higher the order, the sharper the transition. The order must be an integer greater than 1.

8.10 Highpass Elliptical Filter

When the highpass elliptical filter is selected, the window in Figure 15 is displayed. The highpass elliptical filter[9] uses the lowpass elliptical filter techniques described in Section 8.7. However, the highpass filter is defined as a function of the lowpass filter:

$$h_{highpass}(x, y) = 1 - h_{lowpass}(x, y)$$

CutoffM and Cuttoffm: The variables *cutoffM* and *cutoffm* are the cutoff frequencies on the major and minor axes respectively, and lie within the

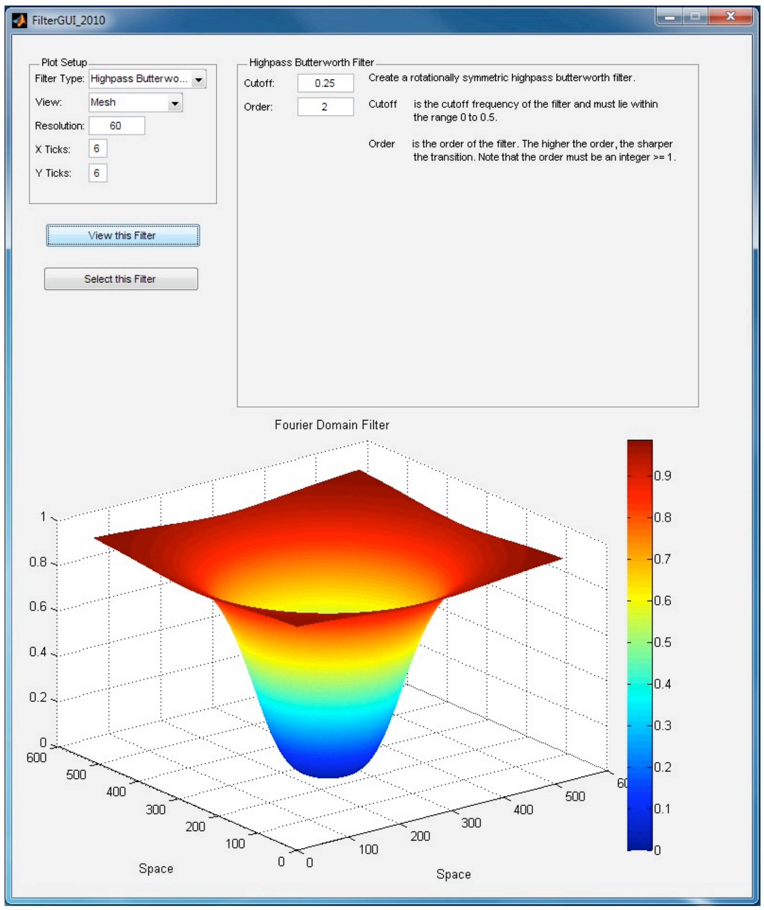

Figure 14: Filter Interface: Highpass Butterworth

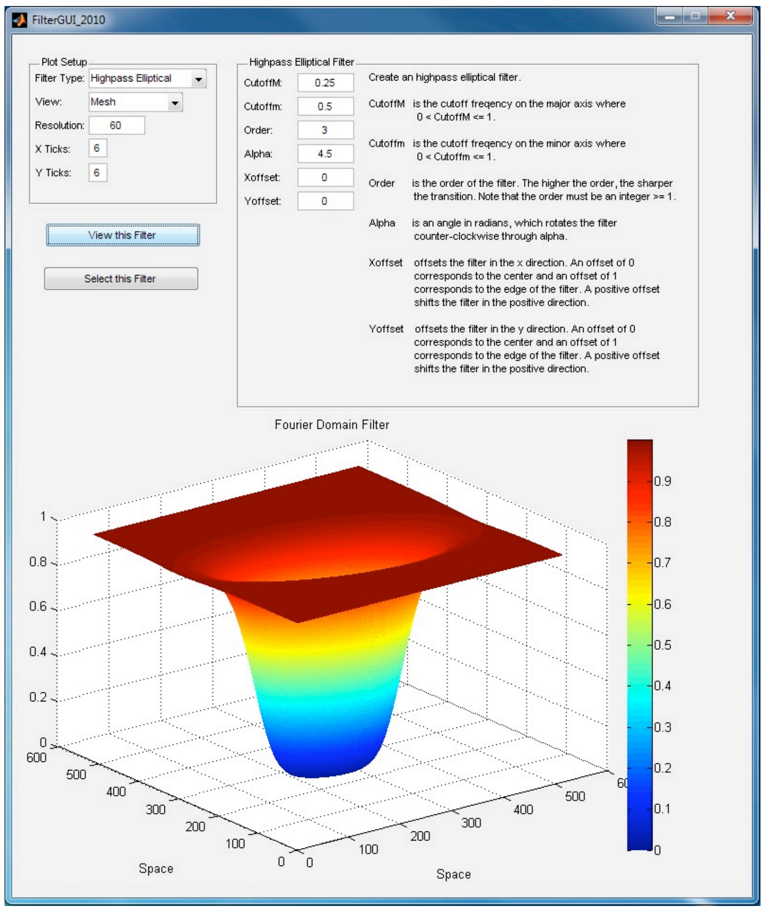

Figure 15: Filter Interface: Highpass Elliptical

range $0 < \mathrm{cutoff} \leq 1$.

Order: *Order* is the order of the filter. The higher the order, the sharper the transition. The order must be an integer greater than 1.

Alpha: The variable *alpha* is an angle in radians, which rotates the filter counter-clockwise through alpha. Obviously and angle of zero does not rotate the filter.

Xoffset and Yoffset: The *Xoffset* and *Yoffset* variables offset the filter in the x and y directions respectively. An offset of 0 corresponds to the center of the filter, and an offset of 1 corresponds to the edge of the filter. A positive offset shifts the filter in the positive direction while a negative offset shifts the filter in the negative direction.

8.11 Bandpass Filter

When the bandpass filter is selected, the window in Figure 16 is displayed. The user can design a bandpass butterworth filter[5] using the following variables:

Cut-in: The *cut-in* variable is the first frequency defining the band pass where $0 < \text{cut-in} \leq 0.5$.

Cutoff: The *cutoff* variable is the second frequency defining the band pass where $0 < \mathrm{cutoff} \leq 0.5$.

Order: *Order* is the order of the filter. The higher the order, the sharper the transition. The order must be an integer greater than 1.

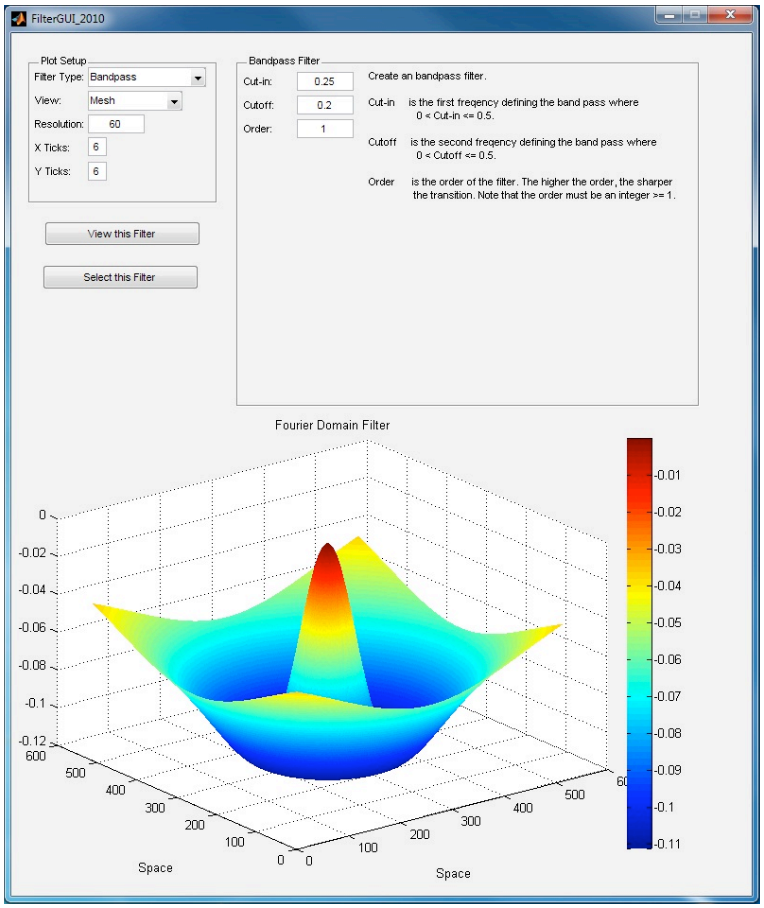

Figure 16: Filter Interface: Bandpass

8.11.1 Alternative Designs Using the Bandpass Filter

Due to the nature of the bandpass filter, a *Lowpass Gaussian* filter can be designed if $\mathrm{cut\text{-}in} = 0$, $\mathrm{cutoff} > 0$, and $\mathrm{order} = 1$. Similarly, a *Highpass Gaussian* filter can be designed if $\mathrm{cut\text{-}in} > 0$, $\mathrm{cutoff} = 0$, and $\mathrm{order} = 1$.

When $\mathrm{cut\text{-}in} > \mathrm{cutoff}$ a *bandpass* filter is designed. When $\mathrm{cut\text{-}in} < \mathrm{cutoff}$ a *bandstop* (or notch) filter is designed.

8.12 Highboost Filter

When the highboost filter is selected, the window in Figure 17 is displayed. The user can design a highboost butterworth filter[6] using the following variables:

Cutoff: The *cutoff* variable is the second frequency defining the band pass where $0 < \mathrm{cutoff} \leq 0.5$.

Order: *Order* is the order of the filter. The higher the order, the sharper the transition. The order must be an integer greater than 1.

Boost: The *boost* variable is the ratio that high frequency values are boosted relative to low frequency values. If *boost* is less than one, than a low-boost filter is generated.

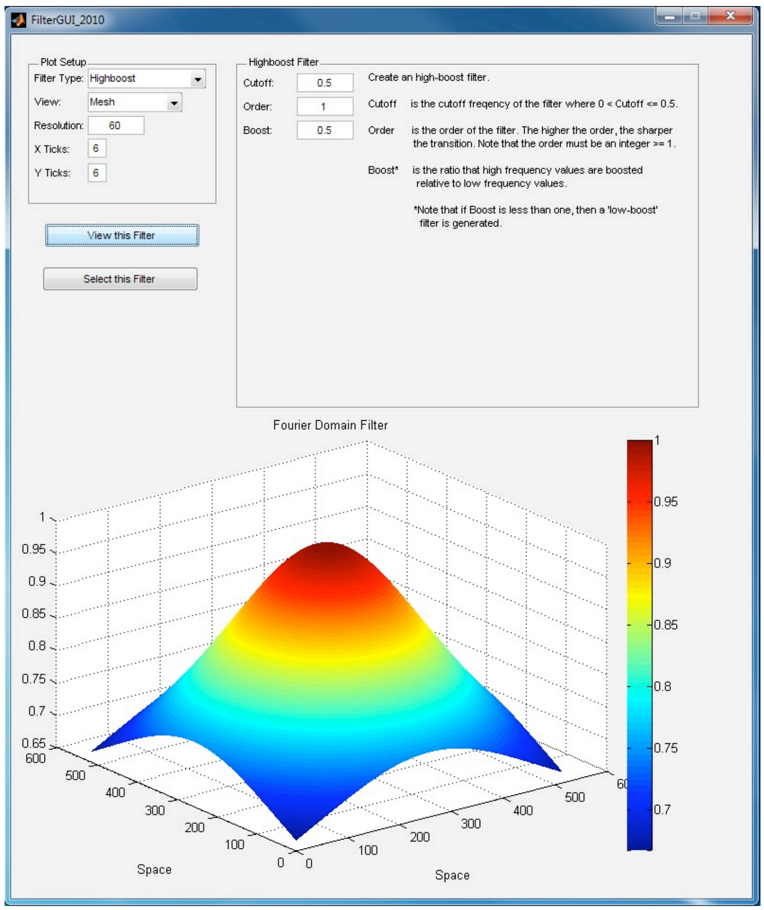

Figure 17: Filter Interface: Highboost

9 Data Processing and Sequence

In order to correctly analyze and interpret results, it is important to fully understand the data processing order and sequence of applying analytical tools. They are as follows.

1. Remove Static Component: MIDAS 2010 checks if the user selected the *Remove Static Component* option, and if so the program determines which static component the user wants to remove; either *Steady State* or *Mean Spectrum*. The definitions of both static components are described in Section 7.2.1. MIDAS 2010 selects all the raw spectrum data originally from the input text file and subtracts the static trace column by column. If the user does not select the *Remove Static Component* option, MIDAS 2010 selects all the raw data originally from the input text file and does nothing to it.

2. Apply Pre-Filter: MIDAS 2010 checks if the user selected the *Apply Pre-Filter* option, and if so the program determines which filter was last designed by the user using the *Pre-Filter Interface*. The pre-filter is applied to the data following the *Remove Static Component* step. Recall from Section 8 that the filter is applied to the data in the Fourier domain. The filter application steps are described in detail in Section 8.2.

3. Compute Average: MIDAS 2010 checks if the user selected the *Aver-*

age option, and if so the program determines which average type the user wants to apply; either *Kernel* or *Dynamic Average*. The averaging technique is applied to the data following the *Apply Pre-Filter* step. The averaging techniques are described in detail in Section 7.2.3. Recall that if the user selects *Dynamic Average* with $\epsilon > 1$, the resulting data matrix will be compressed along the dynamic variable. Thus, if the original data matrix has dimensions $[a\ b]$, namely b points corresponding to the dynamic variable, and a dynamic average is taken every ϵ points, then the resulting data matrix has dimensions $[a\ b/\epsilon]$.

4. Apply Post-Filter: MIDAS 2010 checks if the user selected the *Apply Post-Filter* option, and if so the program determines which filter was last designed by the user using the *Post-Filter Interface*. The post-filter is applied to the data following the *Compute Average* step. Recall from Section 8 that the filter is applied to the data in the Fourier domain. The filter application steps are described in detail in Section 8.2.

5. Timestep: MIDAS 2010 re-computes the temperature (or time) increment between successive points in the data matrix using δ which was defined in the loading window and in Section 5.3. If the user selected *Dynamic Average* with $\epsilon > 1$ in the *Compute Average* step, then the resulting data matrix is compressed along the dynamic (temperature or time) variable; the increment between the columns of the data matrix is no longer δ. Rather, the increment between the columns of

the compressed data matrix is now $\delta * \epsilon$.

6. Choose Selected Data Ranges: MIDAS 2010 now takes the desired ranges into account. The program crops the data matrix resulting from the *Apply Post-Filter* step twice: once along the dynamic variable columns and another along the wavenumber rows. That is, the data between and including *Start X* and *End X*, and *Start Y* and *End Y* is selected and the remaining data is ignored.

7. Special Scales: MIDAS 2010 checks if the user selected the *Scale* option, and if so the program determines which scale type the user wants to apply; either *Regular*, *Variance*, or *Range*. The scaling technique[4] is applied to the data following the *Choose Selected Data Ranges* step. The averaging techniques are described in detail in Section 7.3.3.

8. Fast Fourier Transform and Correlation: Using the data matrix resulting from the *Special Scales* step, MIDAS 2010 takes the fast Fourier transform of the data along the dynamic variable and then performs the univariate correlation. The resulting correlation matrix is then multiplied by 2 to correct to the number of degrees of freedom[4].

   Synchronous Component: The *Synchronous* component is obtained by taking the real part of the resulting correlation matrix.

   Asynchronous Component: The *Asynchronous* component is obtained by taking the imaginary part of the resulting correlation matrix.

Phase Component: The *Phase* component is obtained by computing the phase between the real and imaginary parts of the resulting correlation matrix.

$$\arctan \frac{asyn}{syn}$$

9. Plotting: The viewing options are now taken into consideration. MIDAS 2010 checks how the use wants to view the data matrix; either *Mesh*, *Contour*, *Mesh+Contour*, *3D Contour*, *Contour Fill*, or *Waterfall*. The program plots the data and marks the axes with the appropriate number of tick marks. The desired colour map is also applied (see Section 11.2).

   Spectra: Note that the *Special Scales* and *Fast Fourier Transform and Correlation* steps are not applied to the spectra data. Rather, the remaining data following the *Choose Selected Data Ranges* step is plotted.

In broad terms, the data is processed in the order in which the analytical tools are listed on the main interface.

10 Plotting Buttons

10.1 PLOT Button

Pressing the *PLOT* button will plot or refresh the respective plot while applying the current analytical and viewing settings. The data is processed according to the steps outlined in Section 9.

10.2 Figure Button

When the *Figure* button is pressed, the data is re-processed as outlined in the steps in Section 9. However, rather than being plotted in the appropriate location on MIDAS' main interface, the data is plotted in a separate figure which comes to view. Separating the plot from MIDAS' main interface is advantageous; it grants the user access to Matlab's regular tools. Matlab treats this new figure as any other figure. That is to say, while the plot remains the active figure, the user can enter commands in Matlab's Command Window to further alter the plot. Additionally, since the plot is isolated from MIDAS' main interface, it can be saved, exported, printed, etcetera, as desired.

10.3 Apply Selection to All Button

When the *Apply Selection to All* button is pressed, MIDAS 2010 copies the ranges selected for Plot A with *Start X*, *End X*, *Start Y*, and *End Y*, and

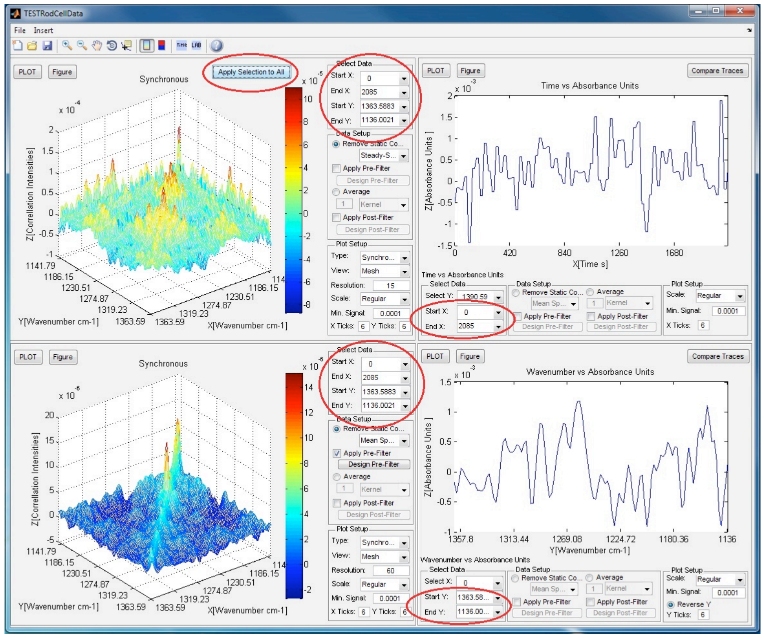

Figure 18: Apply Selection to All

applies these ranges to all plots in the main interface. That is, the ranges for Plots B, 1, and 2 are changed to the ranges selected in Plot A. Then, all plots are refreshed while applying the current analytical and viewing settings of each plot. The data is processed according to the steps outlined in Section 9.

10.4 Compare Traces Button

The *Compare Traces* buttons are only available for Plots 1 and 2. They are tools enabling the user to compare more than one trace at a time. When *Compare Traces* is pressed, the respective plot is refreshed while applying

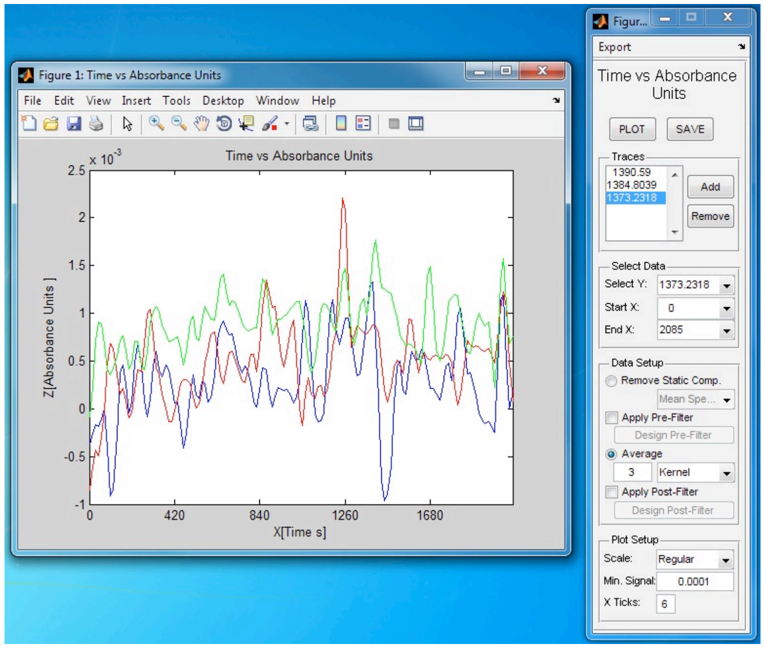

Figure 19: Compare Traces

the current analytical and viewing settings. However, rather than being plotted in the appropriate location on MIDAS' main interface, the data is plotted in a separate figure which comes to view. The data is processed according to the steps outlined in Section 9. This new figure works in conjunction with the trace interface which also comes to view.

Notice how the trace interface is nearly identical to the Plots 1 and 2 sections on the main interface. The essential difference is that the trace interface includes a new *Traces* sections. Using the *Select Y* or *Select X* drop down lists from the *Select Data* section, the user can select which traces she or he would like to compare. Once the trace is selected, press the *Add*

button to add the selected trace to the list. Traces can easily be removed in a similar manner. After selecting the trace from the *Traces* list, press the *Remove* button to remove the trace from the list.

Pressing the *PLOT* button will plot or refresh the conjoining plot while applying the current analytical and viewing settings. All the traces listed in the *Traces* list will be plotted on the conjoining plot. The data is processed according to the steps outlined in Section 9.

Separating the trace plot from MIDAS' main interface is advantageous; it grants the user access to Matlab's regular tools. Matlab treats this new figure as any other figure. That is to say, while the plot remains the active figure, the user can enter commands in Matlab's Command Window to further alter the plot. Additionally, since the plot is isolated from MIDAS' main interface, it can be saved, exported, printed, etcetera, as desired.

10.4.1 Property Editor

Once the conjoining trace plot appears, the user is free to modify the line colours, line styles, line widths, marker types, marker sizes, and marker colours for each trace. To do this, select the *Edit Plot* icon (looks like a mouse arrow) on the trace plot to edit the plot. Then, double click on a trace and the Property Editor will appear at the bottom the the trace plot. The user is free to change any of the line or marker properties at will. Note that the if the user changes the *Plot Type* and refreshes the plot, Matlab is likely to complain because MIDAS 2010 did not account for a change in

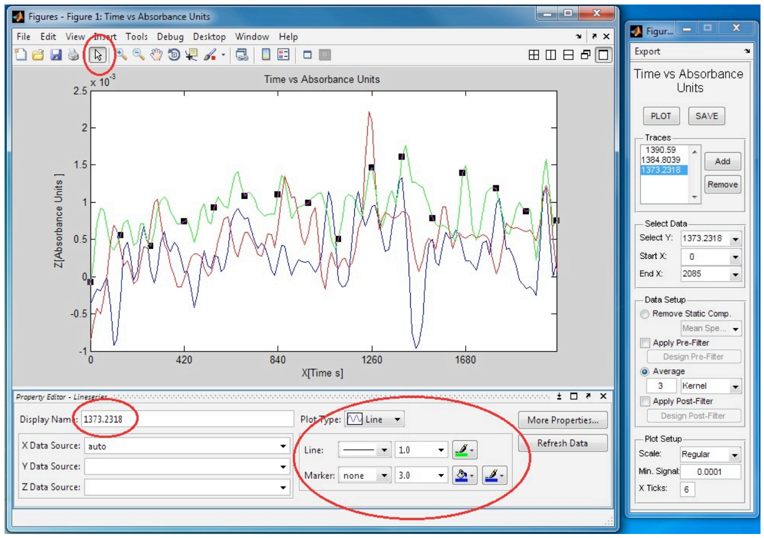

Figure 20: Compare Traces: Property Editor

plot type. Notice how the selected trace is identified by its *Display Name*.

10.4.2 SAVE Button

Pressing the *SAVE* button allows the user to save the conjoining trace plot as an image or as a Portable Document Format (PDF). The saving interface comes to view and assists the user with the saving process. The trace figure can also be saved using the save button from the trace figure itself.

10.4.3 Export Toolbar

There are two exporting options available to the user. The user can export the traces listed in the *Traces* list either as a tab delimited text file or to an Excel file. The file is saved in a location specified by the user.

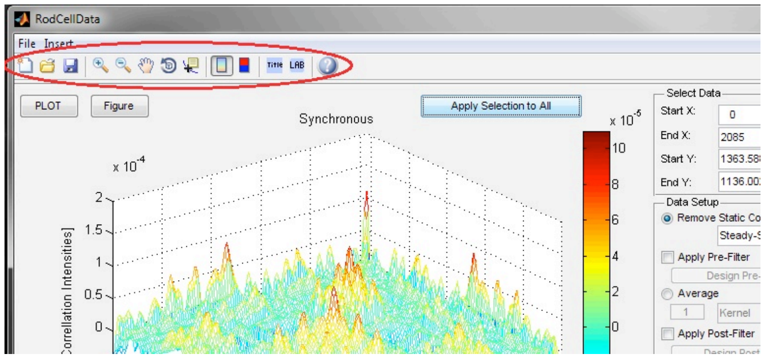

Figure 21: MIDAS 2010 Toolbar

11 Toolbar

MIDAS' 2010 main interface is equipped with a toolbar, where some tools are similar to Matlab's familiar tools and others are unique to MIDAS. The toolbar is identified in Figure 21.

11.1 General Tools

The first eight tools on the toolbar are considered general because they are similar to Matlab's familiar tools. The first three are the *New File*, *Open File*, and *Save File As*.

New File: When the *New File* tool is selected, the loading window shown in Figure 4 appears and assists the user in selecting another file to load for analysis.

Open File: When the *Open File* tool is selected, the user can select and open an existing figure as it was last saved. This tools allows users to

re-open previously saved work for further analysis.

Save File As: When the *Save File As* tool is selected, the user can save the current figure and workspace in a specified location. This tools allows users to save a current work session so that it can be re-openned later on for further analysis. Note that two files are saved when this tool is used: *filename.fig* and *filename.mat* are two files needed to restore previously saved work. Both files are required when the current work session is re-openned at a later date.

11.1.1 Tools For the Active Plot

The next set of tools assist the user in modifying the view of the active plot. A plot is *active* as long as it is the last one plotted by Matlab. That is, when the *Apply Selection to All* or *Blanket Filters* (see Section 12.3.1) are used, Plot 2 is the active plot because MIDAS 2010 refreshes the plots on the main interface in the following order: Plot A, Plot B, Plot 1, then Plot 2. If any particular plot is refreshed using the *PLOT* button, then it is now the active plot.

Zoom In and Zoom Out: The *Zoom* tools are identical to Matlab's zoom tools. Since zooming in and out can be difficult to see within the main interface, it is suggested to press the *Figure* button first to plot the active plot in it's own figure, independent of the main interface. Double clicking on the plot while the tool is activated resets the plot to its original view. Selecting the same tool from the menu will turn the selected tool off.

Pan: The *Pan* tool is identical to Matlab's pan tools. Since panning can be difficult to see within the main interface, it is suggested to press the *Figure* button first to plot the active plot in it's own figure, independent of the main interface. Double clicking on the plot while the tool is activated resets the plot to its original view. Selecting the same tool from the menu will turn the selected tool off.

Rotate 3D: The *Rotate 3D* tool is identical to Matlab's rotate 3D tools. Since rotating can be difficult to see within the main interface, it is suggested to press the *Figure* button first to plot the active plot in it's own figure, independent of the main interface. Double clicking on the plot while the tool is activated resets the plot to its original view. Selecting the same tool from the menu will turn the selected tool off.

Data Cursor: The *Data Cursor* tool is identical to Matlab's data cursor tool. The X, Y, and Z axes refer to the temperature (or time), wavenumber, and absorbance units axes, respectively, from the spectra plot as defined in the loading window. Thus, when the synchronous, asynchronous, and phase plots are viewed, the *Y axis* is listed twice since these plots have wavnumbers plotted along two axes. Selecting the same tool from the menu will turn the selected tool off.

11.2 Colour Maps

The selected colour map applies to both Plots A and B. Every time a new colour map is selected, Plots A and B are refreshed with the new colour

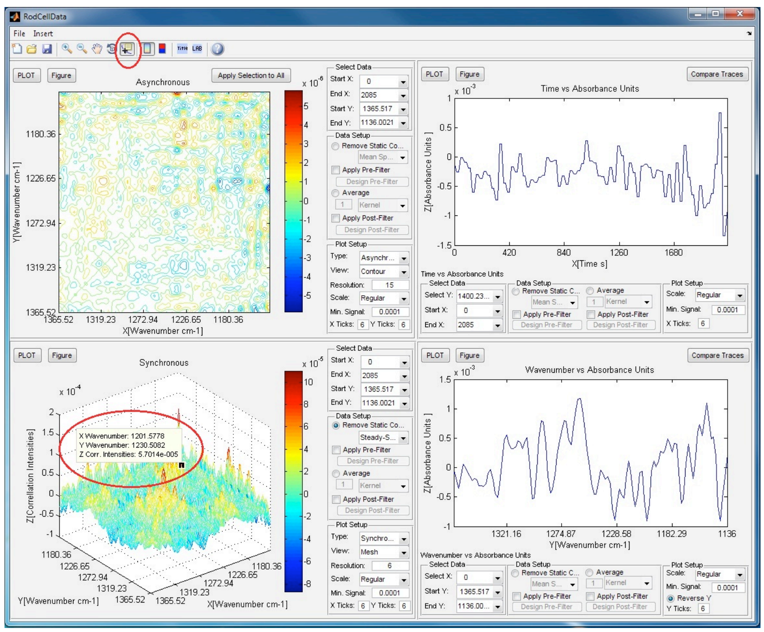

Figure 22: Toolbar: Data Cursor

map. Only one colour map can be selected at a time.

The *Jet* colour map is designed my Matlab and covers most colours in the spectrum. The *Red + Blue* colour map is specially defined by MIDAS 2010. Its purpose it to help the user easily differentiate positive from negative values on a plot. When a graph is plotted, everything above zero is shown in red while everything below zero is shown in blue on Plots A and B. Figures 23 and 24 illustrate the *Red + Blue* colour map option.

11.3 Change Title

Selecting the *Change Title* tool allows the user to change the title of the active plot (see Section 11.1.1 for the definition of the active plot). The *New Title* window shown in Figure 25 will appear to assist the user change the plot title. Since Plots A and B have the option of plotting four different types of plots, namely the synchronous, asynchronous, phase, and spectra graphs, the plot title will not remain each time the plot is refreshed to a different type of plot. Plots 1 and 2 are cross sections of the spectra plots. Thus, when a new title is given to these plots it will remain every time the plot is refreshed.

11.4 Change Labels and Units

Selecting the *Change Labels* tool allows the user to change the axes labels and units. The *Labels and Units* window shown in Figure 26 will appear

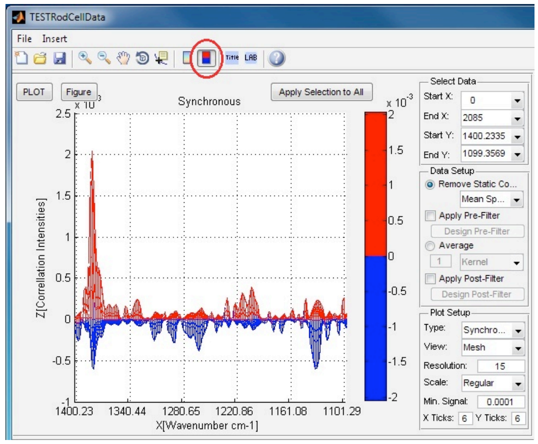

Figure 23: Red and Blue Colour Map

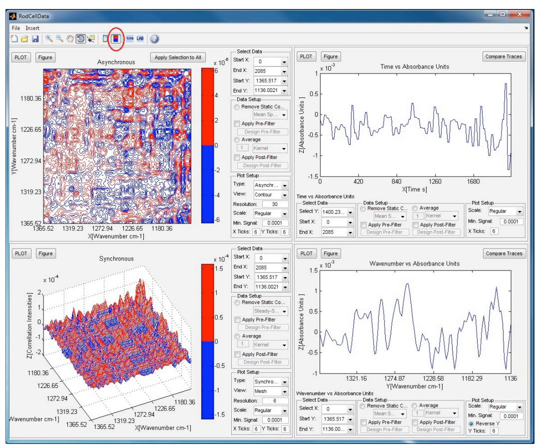

Figure 24: Contour and Rotated Red and Blue Colour Map

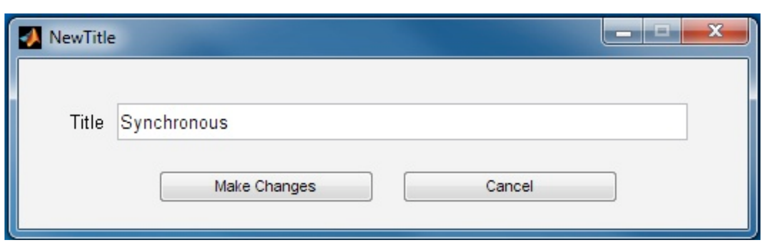

Figure 25: New Title Window

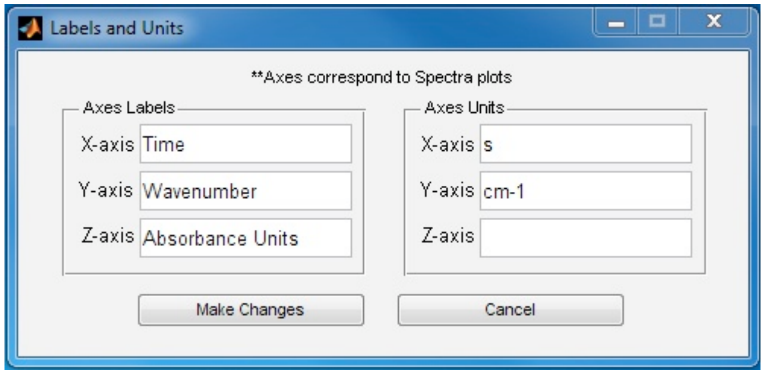

Figure 26: Labels and Units Window

to assist the user change the labels and units. Plots A, B, 1, and 2 will automatically refresh in that order with the new labels and units. Notice that the panel headings for Plots 1 and 2 are also changed to account for the new labels and units. Changing the labels and units with this interface overrides the initial labels and units specified with the loading window.

11.5 Help and User's Guide

Pressing the *MIDAS User Manual* button will automatically open this user guide as a Portable Document Format (PDF).

12 Menu Items

There are two menu items on the top left hand corner of the main window in MIDAS 2010:

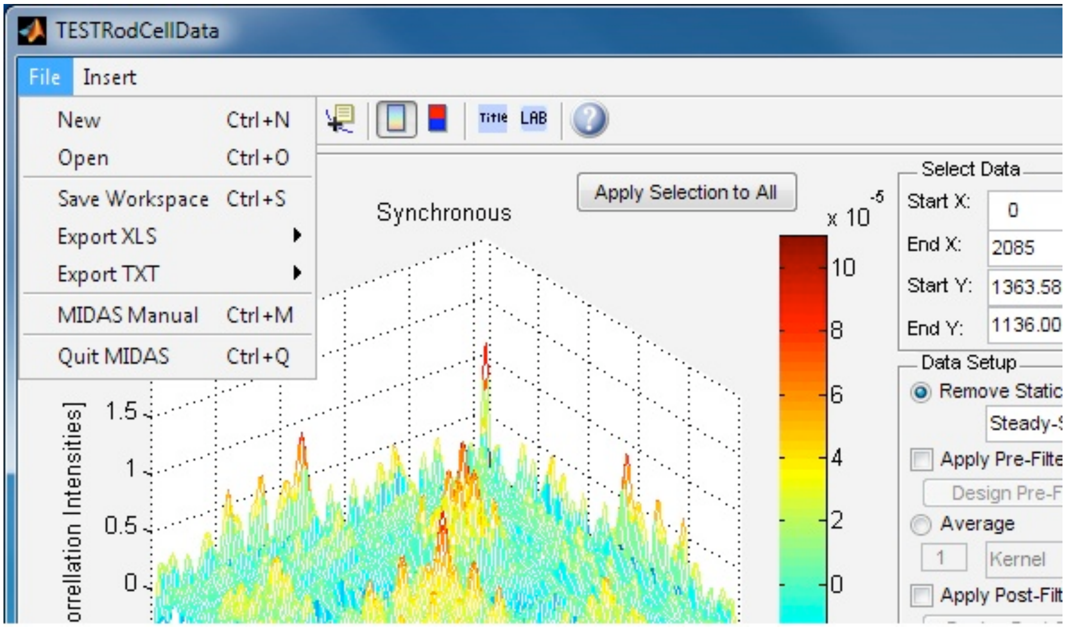

Figure 27: File Menu

- *File*
- *Insert*

They are discussed in detail in the following sections.

12.1 File Menu

The *File* menu is shown in Figure 27.

12.1.1 New and Open

The *New* and *Open* file menu items operate similarly to the *New File* and *Open File* buttons on the toolbar. To load a new set of data in MIDAS

2010, select *New*. The loading window in Figure 4 will appear. To open previously saved work, select *Open*. A dialogue box will appear asking the user to find and select the workspace figure. Once the figure is selected, the workspace window will appear as it was last saved.

12.1.2 Saving Current Work Sessions

The *Save* file menu item allows the user to save figures and current sessions. This will save the current workspace figure and the *.mat* file that is needed to save the data and work done to each plot area. In order to re-open the an existing file, MIDAS 2010 must locate the *.fig* and *.mat* files so it is important that both the *.fig* and *.mat* files are saved. A file can only be re-opened if the workspace is saved. The saving window shown will appear to assist the user save the figure appropriately.

12.1.3 Exporting Data

MIDAS 2010 can export the data used to plot graphs Plot A, B, 1, or 2. After plotting the desired graph, the user can export the data to an Excel spreadsheet by selecting *Export XLS* or to a tab delimited text file by selecting *Export TXT* from the file menu. Data exported into an Excel spreadsheet is sent to a *.xls* file while data exported into a tab delimited text file is sent to a *.txt* file.

The data used to graph Plots A and B are matrices. Therefore, the data sent to the Excel file are also a matrices. The exported data will have the

Y axis down the rows and the X axis across the columns. That is, the front corner of the mesh graph is at top left hand corner of the matrix in the spreadsheet.

When exporting the matrices to graph Plots A and B into a text file, MIDAS 2010 exports the plot as shown on the main interface. That is to say, if the current plot is the synchronous plot, MIDAS 2010 exports the synchronous data within the selected data ranges. If the current plot is the spectra, the program exports the spectra within the selected data ranges. The first column in the text file corresponds to the variables, like wavenumbers, and the rest of the matrix is the data used to plot the graph (ie. the synchronous or spectra matrix).

Since the data for Plots 1 and 2 are simple cross sections of the averaged data, the data is only a set of vectors and not a matrix. Therefore, the data sent to the Excel sheet is two columns: one contains the X and the other contains the Y data points for the plot.

Note that the data sent to a text tile will look very messy because the columns and rows will no longer be properly aligned. Although this is not too appealing to us, the computer knows what data is in each row and in which column. The data should already be tab delimitated; this should be apparent. Do not modify the exported data in the text file.

Important Note: When exporting to a file that already exists, the computer will ask the user if he or she would like to replace the existing file. If the user chooses to replace the existing file, note that the program does

not erase the file completely before writing to it. Rather, it writes over top the old file but does not erase the existing data. Therefore, the user will end up with a mix of data overlapping between old and newly- exported data. To avoid this mishap, be sure to export to a new file with a unique name.

Please see Section 12.2 for details outlining the worksheets created when exporting to XLS. Matlab will issue Warnings saying it added a specified worksheet to the Excel file, if the worksheet does not already exist. This is completely normal. The last message will indicate if the exporting is done:

$$>> Done\ exporting\ to\ XLS!$$

12.1.4 MIDAS Manual

Selecting the *MIDAS Manual* menu item will automatically open this user guide as a Portable Document Format (PDF).

12.1.5 Quit

Before quitting MIDAS from the file menu, the user has the option of saving the current workspace. A dialogue box will appear asking if the user would like to save the workspace. Pressing the X on the top right hand side of the main window will also present the dialogue box.

12.2 Exporting Data to XLS

12.2.1 Information Worksheet

The *Information* worksheet shows the metadata at the time the export was issued.

The labels and units correspond to the axes labels and units specified with the loading window. *DeltaX* is the constant δ temperature (or time) increment between the columns of the data matrix, as explained in Section 5.3. *DeltaY* is the increment between the rows of the data matrix, which is assumed to be constant.

The *StartX*, *EndX*, *StartY*, and *EndY* are the settings of the selected data at the time the export was issued. The settings of the analytical tools are also listed.

The *filepath* and other path names indicate the name of the data file and its location within the computer.

12.2.2 All Spectra Worksheet

The content on the *All Spectra* worksheet is what MIDAS 2010 read as input from the provided text file. The first column of the matrix contains the variables, such as wavenumbers, while the remaining columns contain samples, like absorbance units. The dynamic variable, such as temperature or time, is incremented by a constant amount δ between each column starting from column 2 to n. That is, each row contains the wavenum-

ber in the first cell and the various absorbance data are in the remaining cells. Temperature (or time) is incremented between each column starting with temperature (or time) $T = 0$ in column 2, $T = \delta$ in column 3, ..., and $T = (n - 2) * \delta$ in column *n*.

12.2.3 Selected Spectra Worksheet

The content on the *Selected Spectra* worksheet takes the selection ranges into account by selecting the spectrum data ranging between *StartX* and *EndX*, and *StartY* and *EndY* inclusively. The variables, such as wavenumbers, are no longer in the first column. The entire matrix contain the selected samples, like absorbance units. The dynamic variable, such as temperature or time, is incremented by a constant amount δ between each column starting from column 1 to *n*. Notice how *StartX*, *EndX*, *StartY*, and *EndY* are listed at the top of the worksheet.

12.2.4 Wavenumber and Time Worksheets

The column vector on the *Wavenumber* worksheet shows the selected range of the variables, like wavenumbers, between *StartY* and *EndY* inclusively.

The column vector on the *Time* worksheet shows the selected range of the dynamic variable, like temperature or time, between *StartX* and *EndX* inclusively.

12.2.5 Synchronous, Asynchronous, and Phase Worksheets

The content on the *Synchronous* worksheet shows the synchronous matrix following Step 8 in Section 9. It contains the real part of the correlation matrix.

The content on the *Asynchronous* worksheet shows the asynchronous matrix following Step 8 in Section 9. It contains the imaginary part of the correlation matrix.

The content on the *Phase* worksheet shows the phase matrix following Step 8 in Section 9. It contains the phase between the real and imaginary parts of the correlation matrix.

12.2.6 Static Component Worksheet

If the user removed a static component at the time the export was issued, the *Static Component* worksheet contains the static column vector; either the steady state or mean spectrum vector.

12.2.7 Pre- and Post- Filter Worksheets

If the user applied a filter to the data at the time the export was issued, the contents on the *Prefilter* and *Postfilter* worksheets show the pre- and post- filter applied to the data, respectively. Note that the DC component is is the corners of the filter and not in the center of the matrix. Typically, Matlab shifts the filter so the DC component is placed in the center of the

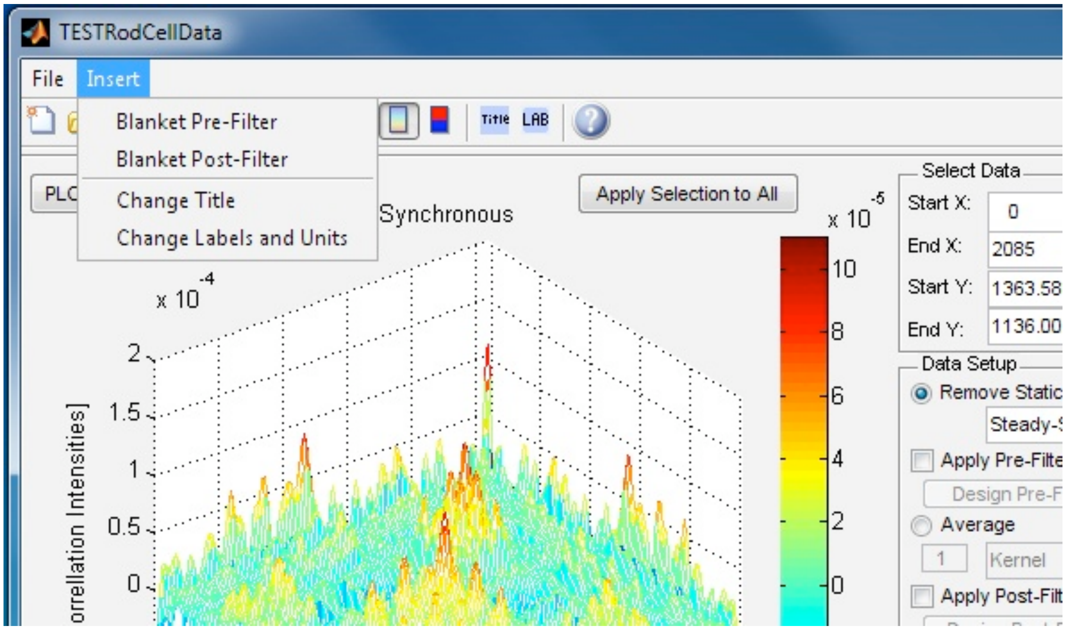

Figure 28: Insert Menu

filter before applying it to the data. Notice how the filter type and its design parameters are listed at the top of the worksheet.

12.3 Insert Menu

The *Insert* menu is shown in Figure 28.

12.3.1 Blanket Filters

The *Blanket Filter* menu item will bring up the filtering window shown in Figure 7. The operations of the filtering window are explained in Section 8 above. When a filter is selected from the *Blanket Filter* menu item, the

designed filter is applied to all four plot areas in the main window. Therefore, Plots A, B, 1, and 2 are refreshed in that order with the newly filtered graphs.

If *Blanket Filter* is selected from the Insert menu, and no filter is designed, meaning the user pressed the X on the top right hand corner of the filtering interface, then no filter is applied to the data. Notice how the *Apply Filter* checkboxes are not checked and Plots A, B, 1, and 2 are refreshed without applying the filter.

12.4 Change Title

Selecting the *Change Title* menu item allows the user to change the title of the active plot (see Section 11.1.1 for the definition of the active plot). The *New Title* window shown in Figure 25 will appear to assist the user change the plot title. Since Plots A and B have the option of plotting four different types of plots, namely the synchronous, asynchronous, phase, and spectra graphs, the plot title will not remain each time the plot is refreshed to a different type of plot. Plots 1 and 2 are cross sections of the spectra plots. Thus, when a new title is given to these plots it will remain every time the plot is refreshed.

12.5 Change Labels and Units

Selecting the *Change Labels* menu item allows the user to change the axes labels and units. The *Labels and Units* window shown in Figure 26 will appear to assist the user change the labels and units. Plots A, B, 1, and 2 will automatically refresh in that order with the new labels and units. Notice that the panel headings for Plots 1 and 2 are also changed to account for the new labels and units. Changing the labels and units with this interface overrides the initial labels and units specified with the loading window.

13 Dealing with Problems

There are a few options available to the user when a problem is encountered using MIDAS 2010. They are discussed in the following sections.

13.1 fspecial Error Messages

If you get an fspecial error message, you do not have the Image Processing Toolbox installed on your computer. Therefore, you cannot use filtering or kernel average options since these options use the *fspecial* function.

13.2 Compare Traces: Property Editor

Note that the if the user changes the *Plot Type* and refreshes the plot, Matlab is likely to complain because MIDAS 2010 did not account for a change

in plot type. Close both the trace interface and conjoining plot to stop Matlab's complaints.

13.3 Plotting Problem

If the user notices MIDAS 2010 incorrectly graphed a plot, simply refresh the plot by pressing the *PLOT* button. It is possible that MIDAS encountered a problem because of the particular selected plotting options. If a warning is issued on Matlab's Command Window, please record it as well as the current plotting options. The programmer should be able to find and repair the problem.

13.4 Stopping Matlab's Computation

Sometimes MIDAS 2010 might take a long time to generate or refresh a plot. This might be because the selected data set is large or because Matlab encountered a problem. To stop Matlab's computations, press *Ctrl C* in Matlab's Command Window. A warning is most likely to appear because Matlab was forced to stop in the middle of an operation.

13.5 Frozen Mouse

If, for some reason, the user does not have a mouse or if the mouse has frozen, MIDAS 2010 can still be manipulated with the keyboard. By pressing the *Tab* button, the user can jump between options. The *Spacebar* is

used as a selector.

13.6 Frozen Computer

When the computer freezes, MIDAS 2010 and Matlab are no longer responding. The programs must be quit. To do this, press *Ctrl* + *Alt* + *Delete* and terminate Matlab. All unsaved work will be lost.

13.7 Contact Information

Elise Normand designed and created MIDAS 2010. Please report all problems and direct all questions to her via email at `elise.normand@usask.ca`.

14 Acknowledgments

Thanks to Ference Borondics and to Tim May at the Canadian Light Source for their support. A special and heartfelt thanks is also extended to Luca Quaroni who was the Staff Scientist on the Mid-Infrared Spectromicroscopy beamline at the Canadian Light Source when MIDAS was initially created. This program would not have been possible without their generous help and guidance.

References

[1] Eramian, Mark. eramian@cs.usask.ca, Associate Professor, Department of Computer Science, University of Saskatchewan, 29 July 2008.

[2] Eramian, Mark. eramian@cs.usask.ca, *CMPT 487/819, Frequency Domain Filtering*, Associate Professor, Department of Computer Science, University of Saskatchewan, October 2009.

[3] Eramian, Mark. eramian@cs.usask.ca, *CMPT 487/819, Preprocessing: Intensity Transformations and Spatial Filtering*, Associate Professor, Department of Computer Science, University of Saskatchewan, October 2009.

[4] Harrington, Peter de B., Urbas, Aaron., and Tandler, Peter J.. *Tutorial: Two-dimensional correlation analysis*, ELSEVIER, Chemometrics and Intelligent Laboratory Systems, 30 October 1999, pages 151-152, 162-163, 167-168.

[5] Kovesi, Peter. pk@cs.uwa.edu.au, *bandpassfilter*, Department of Computer Science and Software Engineering, University of Western Australia, http://www.owlnet.rice.edu/~elec301/Projects01/image_filt/matlab.html, October 1999.

[6] Kovesi, Peter. pk@cs.uwa.edu.au, *highboostfilter*, Department of Computer Science and Software Engineering, University of West-

ern Australia, http://www.csse.uwa.edu.au/~pk/Research/MatlabFns/, November 2001.

[7] Kovesi, Peter. pk@cs.uwa.edu.au, *highpassfilter*, Department of Computer Science and Software Engineering, University of Western Australia, http://www.owlnet.rice.edu/~elec301/Projects01/image_filt/matlab.html, October 1999.

[8] Kovesi, Peter. pk@cs.uwa.edu.au, *lowpassfilter*, Department of Computer Science and Software Engineering, University of Western Australia, http://www.owlnet.rice.edu/~elec301/Projects01/image_filt/matlab.html, October 1999. Modified by Gaddi, Rob. gaddi@rice.edu, ELEC 301, Rice University, December 2001.

[9] Streit, Katie. kstreit@rice.edu, *EFilter*, ELEC 30, Rice University, http://www.owlnet.rice.edu/~elec301/Projects01/image_filt/matlab.html, December 2001.